Chapter 6 Spatial data sharing, data warehousing & DB federation

1. Introduction

spatial data sharing - become commercial business & a standard practice in modern data processing
- in many countries, it’s a cornerstone of the national info infrastructure

2. The concept & method of spatial data sharing

2.1 The definition & nature of spatial data sharing

data sharing aims to make DB sys interoperable at the data, application, business process levels
(→ Tables 6-1)

data sharing include interoperability of DB & integration among different sys thematically + hierarchically

data sharing is dependent decreasingly on the physical transfer of data
increasingly on virtual & logical sharing of data

Table 6-1. Levels of spatial data sharing

Levels of Data Sharing	***Data Sharing Characteristics***			
	Computing Environment	***Systems Architecture***	***Procedure***	***Purposes and Applications***
Infrastructure	Open computing standards (Internet, Web services, Java), distributed processing and distributed objects	A distributed network of databases connected by the global telecommunicat-ions system	Global/Universal information access and application through database mediation and information brokering	Seamless spatial database interoperability and integration using operational and legacy data
Enterprise		Federated databases and data warehouses connected to an organisation's communications network	Inter-departmental information access and application through database mediation and information brokering	Simultaneous on-line transaction processing (OLTP), on-line analytical processing (OLAP)
Domain	Distributed databases connected using TCP/IP, HTTP and open database connectivity standards	Three-tiered client/server computing or data mart in a wide area network (WAN)	Shared databases with sophisticated collaboration among different users or organisations	Sector-based data management and applications, multi-sourced spatial data analysis and modelling
Functional		Two-tiered client/server computing in a local area network (LAN)	Heterogeneous data exchange	Spatial data visualisation and overlay analysis using data from multiple sources
Connected	Peer-to-peer proprietary network and communications protocols	Desktop computer with simple network connection	Homogeneous data exchange	Electronic exchange of text files and graphics files of the same format
Ad Hoc	Stand alone computers and independent data files	Independent desk top computers	Manual data exchange with hard copy maps, diskettes, CD-ROM	Occasional exchange or sale of data from ad hoc requests

2.2 The importance of spatial data sharing

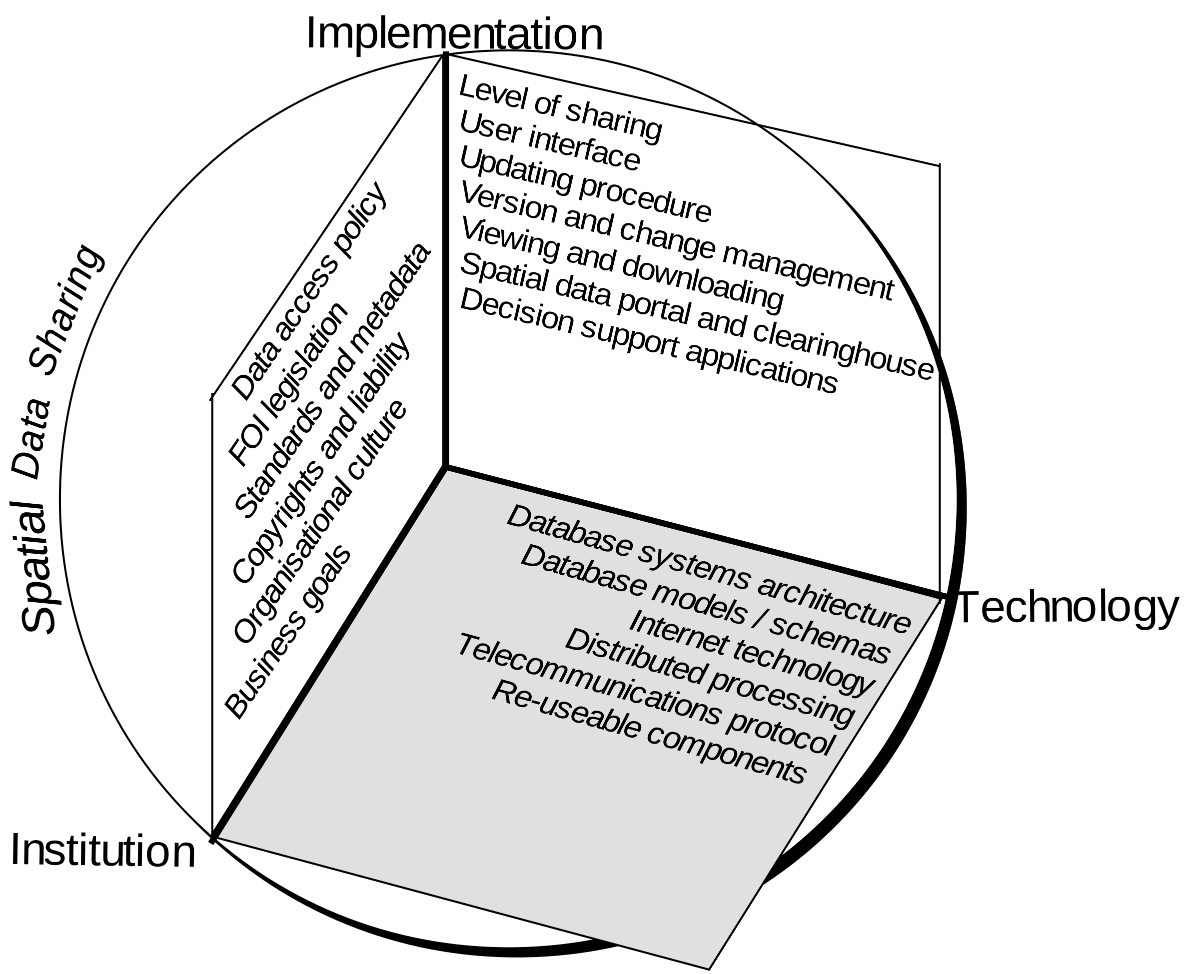


Fig 6-1 Spatial data sharing from the perspective of implementation, technology & institution of DB sys

2.3 Barriers to spatial data sharing

several general barriers to spatial data sharing

a. inherently complex & diverse characteristics of spatial data

b. non-deterministic nature of human cognition & language → non-standardized terminology

c. differences in data management policies, user access protocols, systems security measures, DB partitioning, DB structuring, network bandwidths, HW/SW standards

d. technical incompatibilities & system dependencies

e. lack of data model & format standardization

f. restricted availability of data

g. unwillingness of organizations to share data

h. restriction on releasing data to the public

i. diversity in the types of users & the disparities between the needs of high-end & low-end users

j. lack of coordination between spatial data collectors

k. lack of a supporting data discovery & delivery infrastructure

l. high cost of data

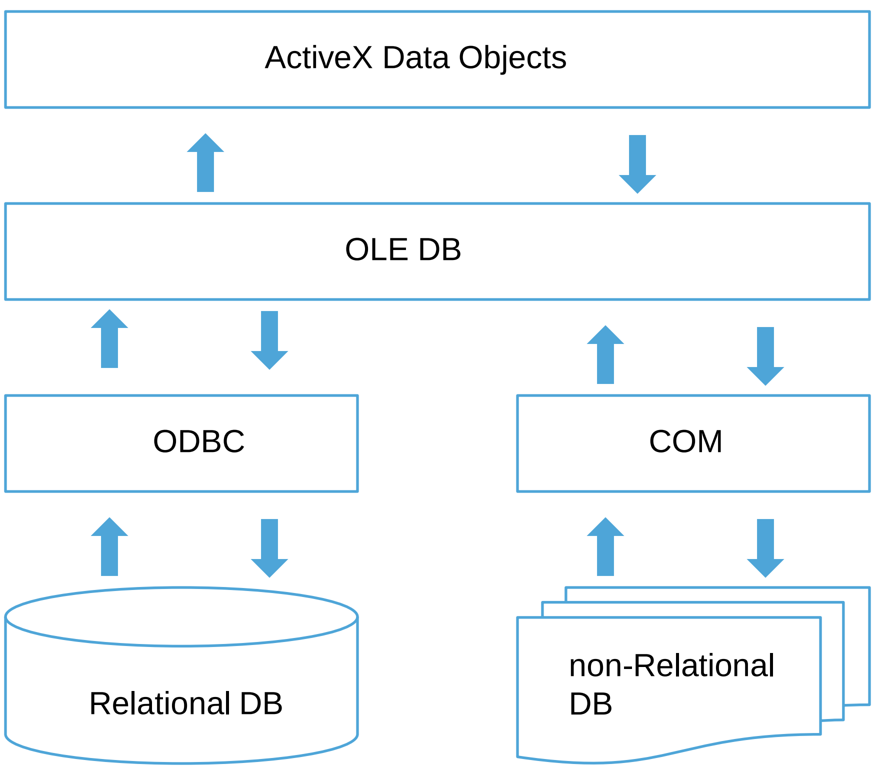
2.4 A standard-based framework for spatial data sharing

users generally endorse the concept of interoperability thru standardization (by OGC)

DB access standard : OLE, ODBC, JDBC, CORBA, COM

web-based services protocols : data sharing over the internet

* OLE DB programming overview



2.4.1 Object linking & embedding (OLE)

de facto standard for data access in MS window environment

common data access method for MS server SW (ex. SQL Server)

+ desktop application SW (PPT, Word, Access, Excel)

+ application development tools (ex. Visual Basic, Visual C++, .NET)

allow users to access & exchange data freely among MS SW products

use COM-based programming interface

a variety of OLE-based technologies have been published (by MS) – ex. OLE DB

* OLE DB - allows an SQL-based data access approach to query a relational / object-relational DB

- allows cross-component business rules & create objects that export an event model

2.4.2 Open DB connectivity (ODBC)

open standard API for accessing DBs from an application

created by SQL-Access Group (SAG) → extended by MS as a standard Call-Level Interface (CLI)

main idea is to use a middle layer (called a DB driver) between the data source and application

- function of driver - translate the queries from an application into commands
- to work, both the data source & applications must be ODBC-compliant
- each data source must have a specific ODBC driver for it to be accessed
  - user install the necessary ODBC drivers from DB SW vendors / 3rd party SW developers

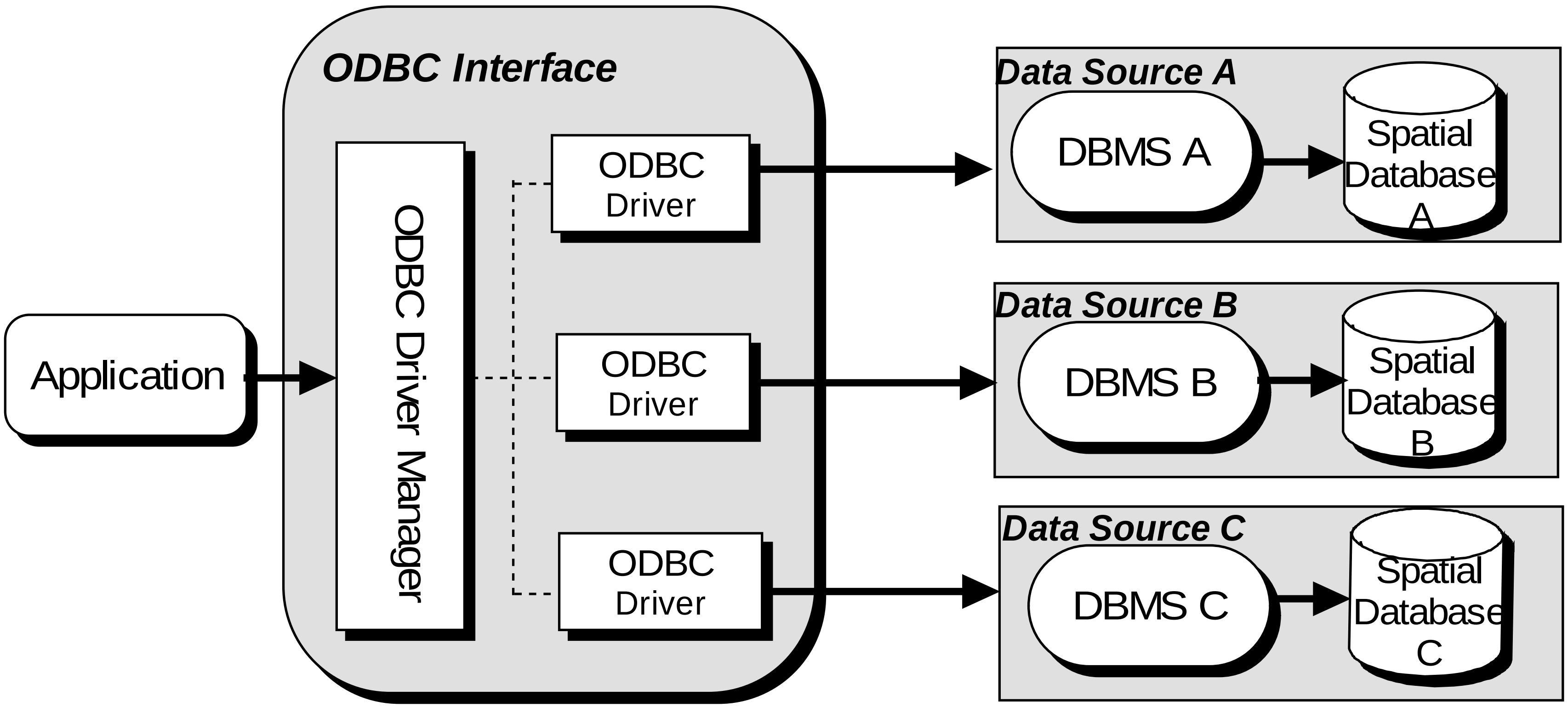


Fig 6-2 The working principles of open DB connectivity (ODBC)

2.4.3 Java DB connectivity (JDBC)

API specification for Java applets, servlets, applications to access data in DB, spreadsheet, flat files

commonly used to connect a user application to a data source regardless of DBMS used

very similar to ODBC function & architecture

can be used w/ a one-tier driver / two-tier driver / a JDBC network driver

can be downloaded from the SUN MicroSystems website

applet

웹에서 사용할 수 없는 코드들로 웹과 분리되어 웹브라우저에 담겨서 실행되는 작은 프로그램

동작은 웹브라우저에 의해 다운로드 된 후 설치되어 실행 (브라우저 내의 JVM에 의해 실행)

실행순서 - 웹브라우저가 처음 읽힐 때 init() 메소드 호출 - 페이지를 사용자가 볼 때마다 start() - 페이지를 떠날 때 stop() - 중간에 프로그램내에서 / 애플릿 화면을 다시 그려야 할 때 paint() 호출

```
import java.awt.*;
 import java.applet.*;
 public class Applet1 extends Applet{     // Applet1클래스는 Applet을 상속하였기 때문에 애플릿임
     public void paint(Graphics g){
         this.setBackground(Color.red);
         g.drawString("애플릿의 세계로...",20,50);
     }
 }
```

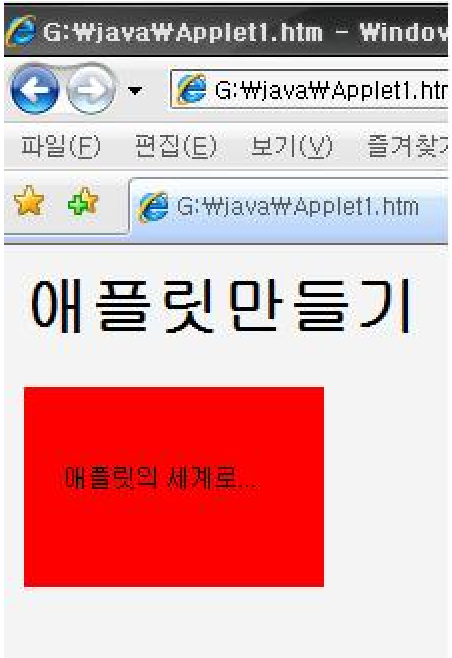
main메소드가 사용되지 않았는데 이것은 애플릿의 특성과 관련
응용 프로그램과 달리 웹브라우저에서 실행되므로 main메소드가 필요 없음

```
<Htm>
<Body>

   <h1>애플릿만들기</h1>

    <Applet code = "Applet1.class" width=150 height=100>
    </Applet>

</Body>
</Htm>
```



웹 페이지를 위와 같이 작성하고 Applet1.htm 으로 저장
(이 때, Applet1.class와 같은 디렉토리 안에 있어야 실행 된다고 함)

servlet

Java를 사용하여 웹페이지를 동적으로 생성하는 서버측 프로그램 / 사양

* 동적이란 : 요청에 대해 서버에서 처리된 내용(ex. DB에서 읽어온 내용)을 변경해서 보여줄 수 있음을 의미

컴파일을 거쳐 웹 어플리케이션에 등록이 되면 JSP와 같은 서비스를 제공

* JSP(Java Server Page) : 동적인 페이지로서 서버 측의 스크립트를 의미

JSP 동작원리와 일반적인 HTML 동작원리

일반적인 HTML의 동작원리

```
                              request
웹브라우저    -------------------------------------------->   Web Server    ----------------------------->   HTML
              <--------------------------------------------                 <-----------------------------
                              response
```

JSP의 동작원리

```
                  request
웹브라우저    ----------------->          Web Server           ------>   (JSP)   ------>  Servlet
              <----------------   (Web Application Server)      <---------------------
                 response
```

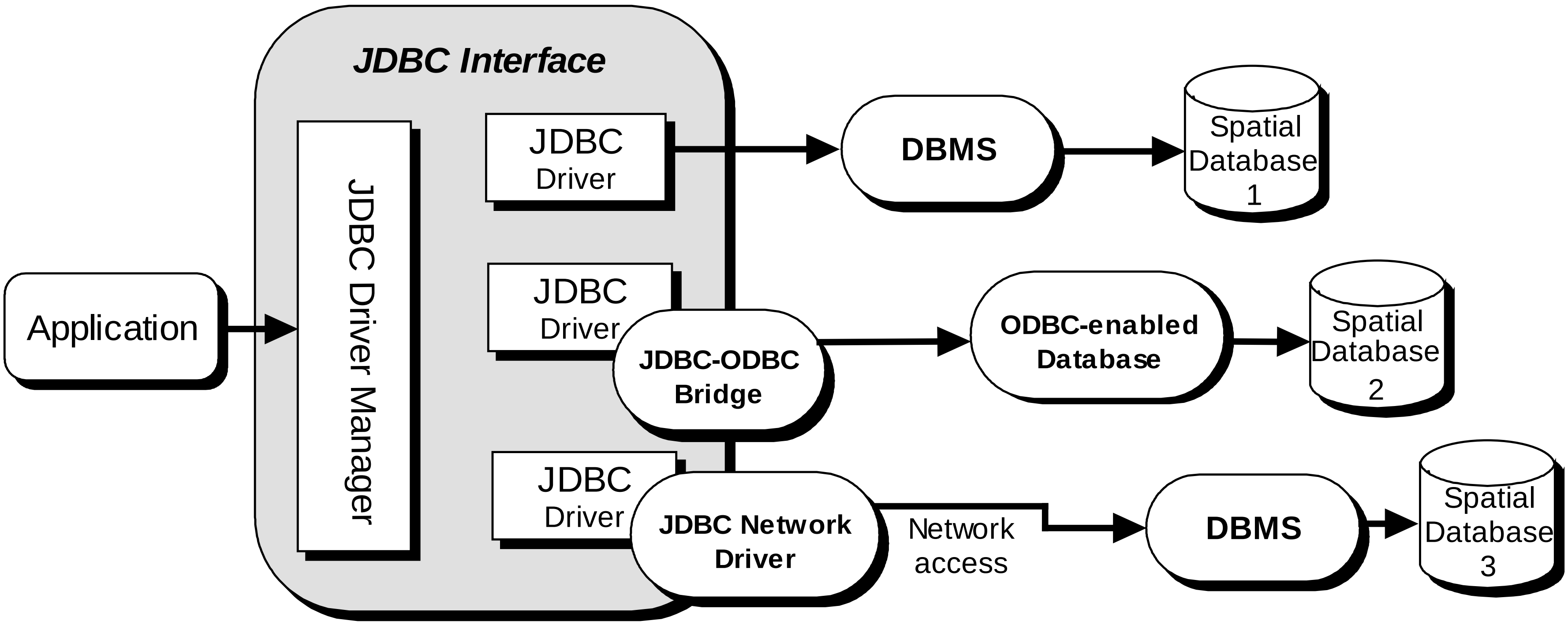


Fig 6-3 The working principles of Java DB connectivity (JDBC)

2.4.4 Common object request broker architecture (CORBA)

ORB - middleware tech that manage communication & data exchange between objects in OOP & DB
provides a directory of services, helps establish connections between these services & clients, activates the required services on behalf of the client, returns the results to the clients

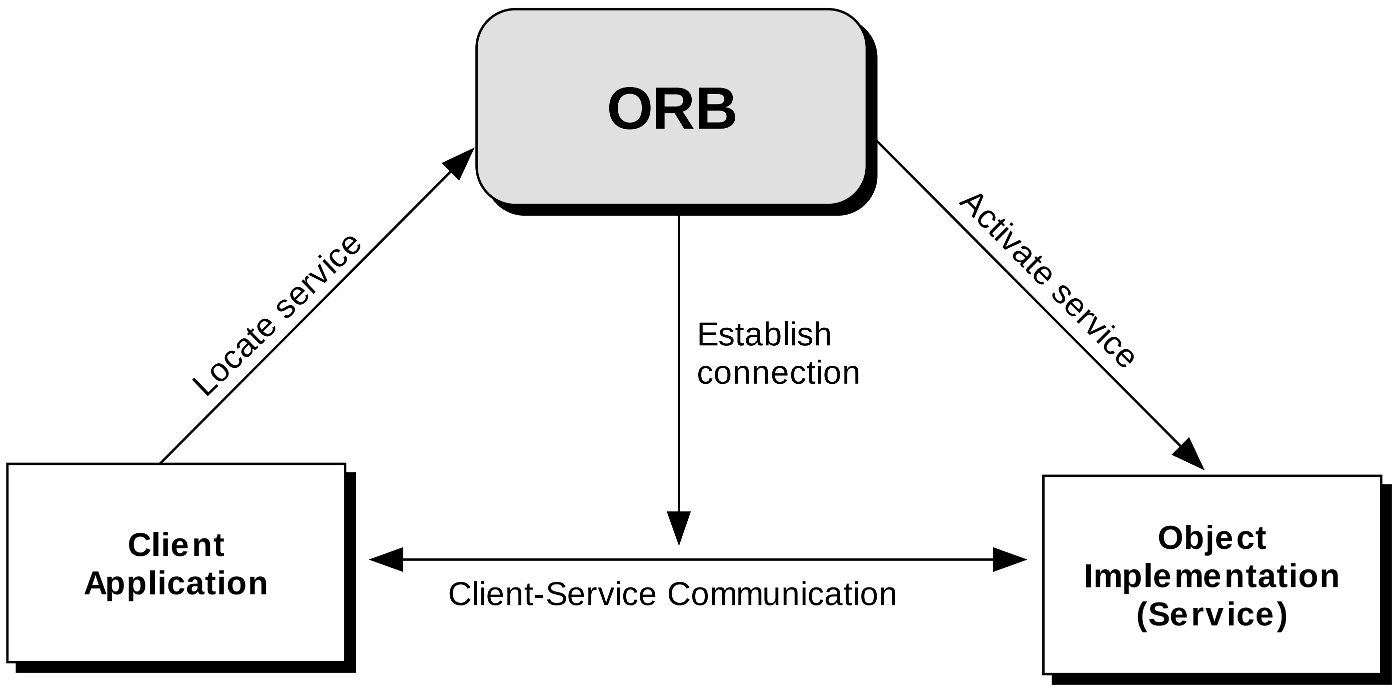


Fig 6-4 The working principles of object request broker

Interfaces & relationships of CORBA

Client : the application program that invokes a method / operation

Object implementation : a service / method of **object (identity + interface + implementation)**

IDL : Interface Definition Language : that needs mapping to programming languages, ex. Java, C++

Interface repository : run-time distributed DB that contains IDL-defined interfaces, serves as a dynamic metadata repository for ORBs

ORB core : run-time infrastructure for transparent communication

DII : Dynamic Invocation Interface : mechanism for constructing requests at run time

Client IDL stub : static interfaces that define how clients invoke corresponding services on the servers (allows to create client-side codes)

ORB interface : an interface containing help functions, APIs that can be used by a client / an object implementation to enhance / extend their functionality

IDL skeleton : server-side analogue of IDL stub - receive requests for services from the object adapter, call the appropriate operation in the object implementation
(allow to create server-side codes)

DSI : Dynamic Skeleton Interface : server-side counterpart of DII

Object adapter : mechanism that assists the ORB w/ delivering requests to the object & activating it

Implementation repository : run-time repository of info about the classes a server supports, the objects that are instantiated, their respective identities

CORBA Java example : http://blog.naver.com/sunsky98/80061571109

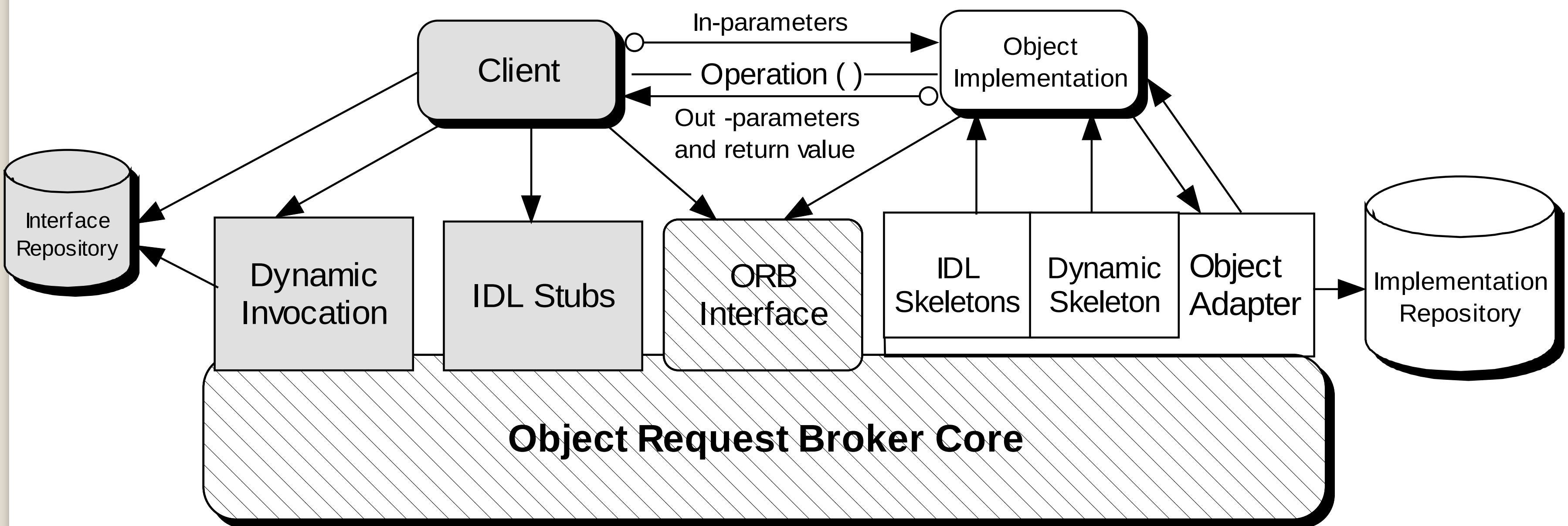


Fig 6-5 Interfaces & relationships of CORBA

2.4.5 The component object model (COM)

ORB specification & implementation by MS

provides a framework for integrating SW components in Window environment

DCOM - extension of COM for network-based interaction between remote clients & services

client applications interact w/ COM components thru interface pointers

SW components interacts in two different ways :

in-process serving : packaging a COM server as a DLL(Dynamic Linking Library) that is loaded into the client process when a class within the server is first accessed by a client

calls made by a client go directly to a component object

out-of-process serving : packaging the COM server as a separate executable running on the same computer / on a remote computer over a network using a DCOM approach

calls go to an object proxy (invokes requests using a remote procedure call)

→ stub receives incoming requests & dispatches it to component objects

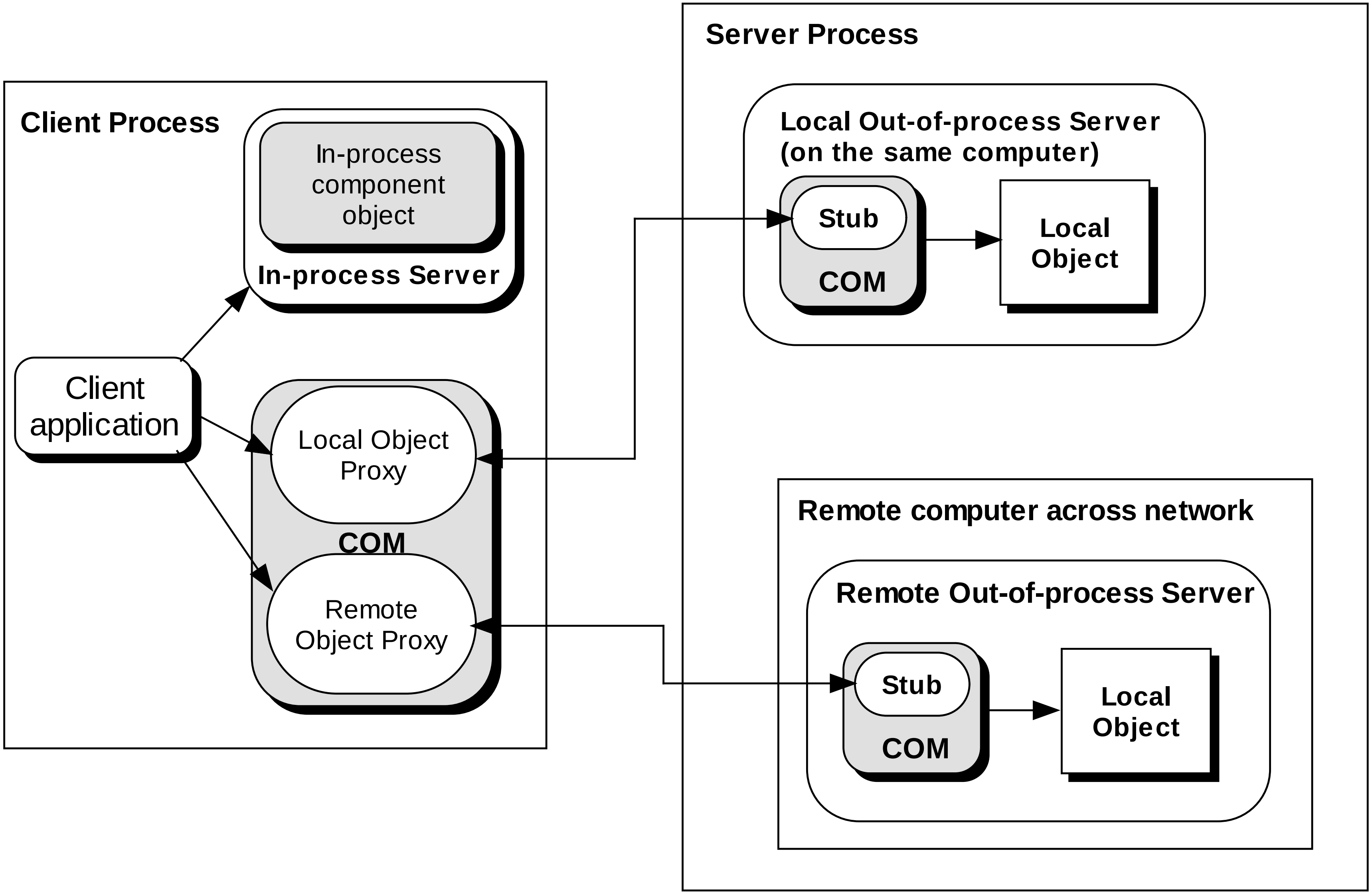


Fig 6-6 The working principles of the common object model & DCOM

2.4.6 Web services protocols

from technical perspective -

SW components that can be accessed over the internet thru standard-based protocols

ex. desktop application programs, web browsers, Java applets, SW running on mobile devices

from application perspective -

standardized way of integrating web-based applications using open standards

web services-based architecture - consists of 3 nodes : client, service, broker

important web services standards

- XML (Extensible Markup Lang) : lang for structuring the messages between nodes
- GML (Geographic Markup Lang) : defines a data encoding in XML for transmission
- SVG (Scalable Vector Graphics) : XML-based format for defining 2D graphics
- SOAP (Simple Object Access Protocol) : standard for transmitting messages between nodes
- WSDL (Web Service Description Lang) : standard lang for describing services
- UDDI (Universal Description, Discovery, Integration) : standard for registering available services & creating a directory service for users to locate & identify available services

OGC developed implementation specifications for the architecture, creation, operation of WMS

* WMS : Web Map Service

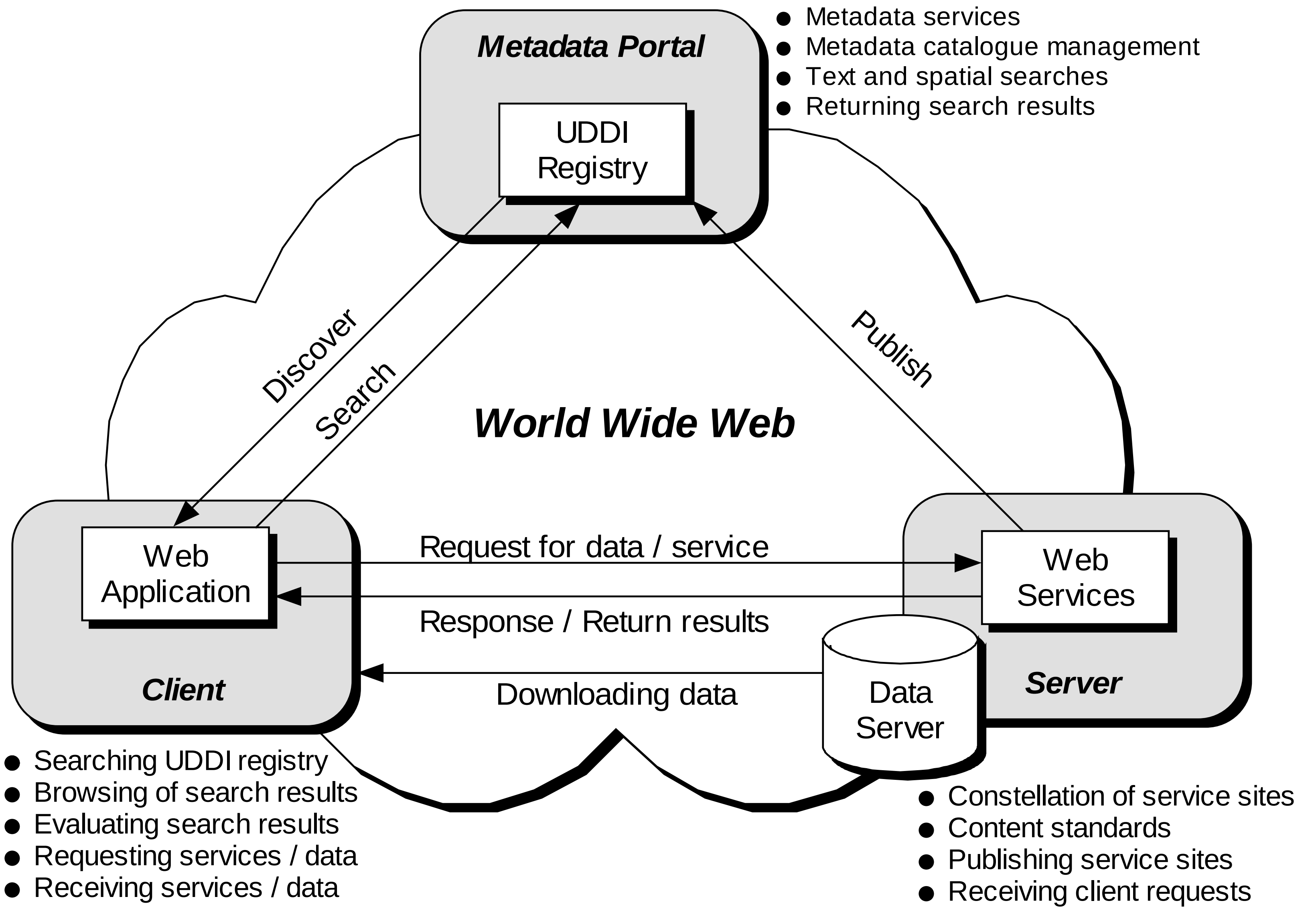


Fig 6-7 The web services-based framework for spatial data sharing

3. DB heterogeneity & its solutions

3.1 The nature & characteristics of DB heterogeneity

typology of heterogeneity - data vs. system heterogeneity

data heterogeneity - 3 aspects

- syntactic h. - due to different data types, storage word lengths, precision, data format, permissible values, units of measurement, abbreviation, acronyms
- structural / schematic h. - due to different data models ex. various vector & raster format, data transfer format, relational/ OO/ O-relational DBs
- semantic h. - due to the inconsistencies / disagreements between digital representation & corresponding real world features
  caused by interplay of factors such as conceptualization h., formalization h., context h.

two conceptual constructs to solve heterogeneity - increase interoperability

- use of ontology
- use of info mediation

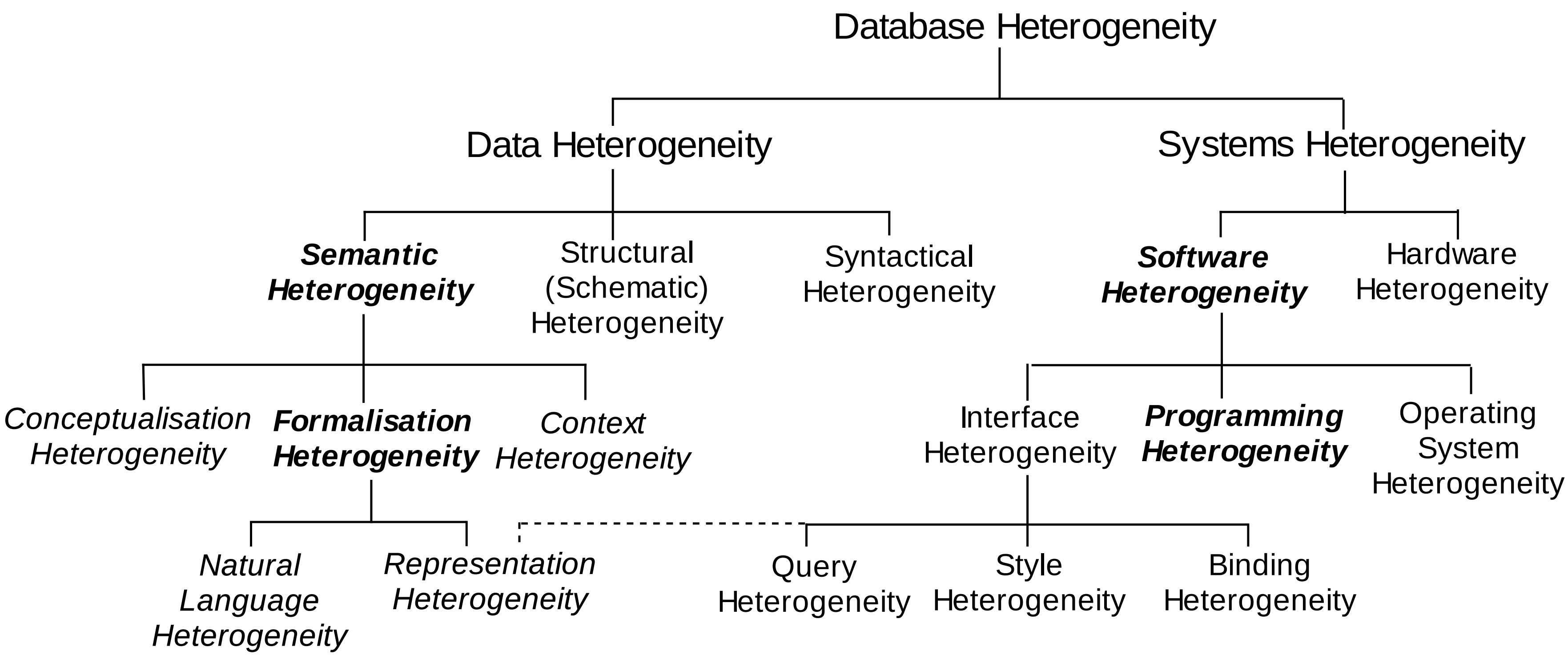
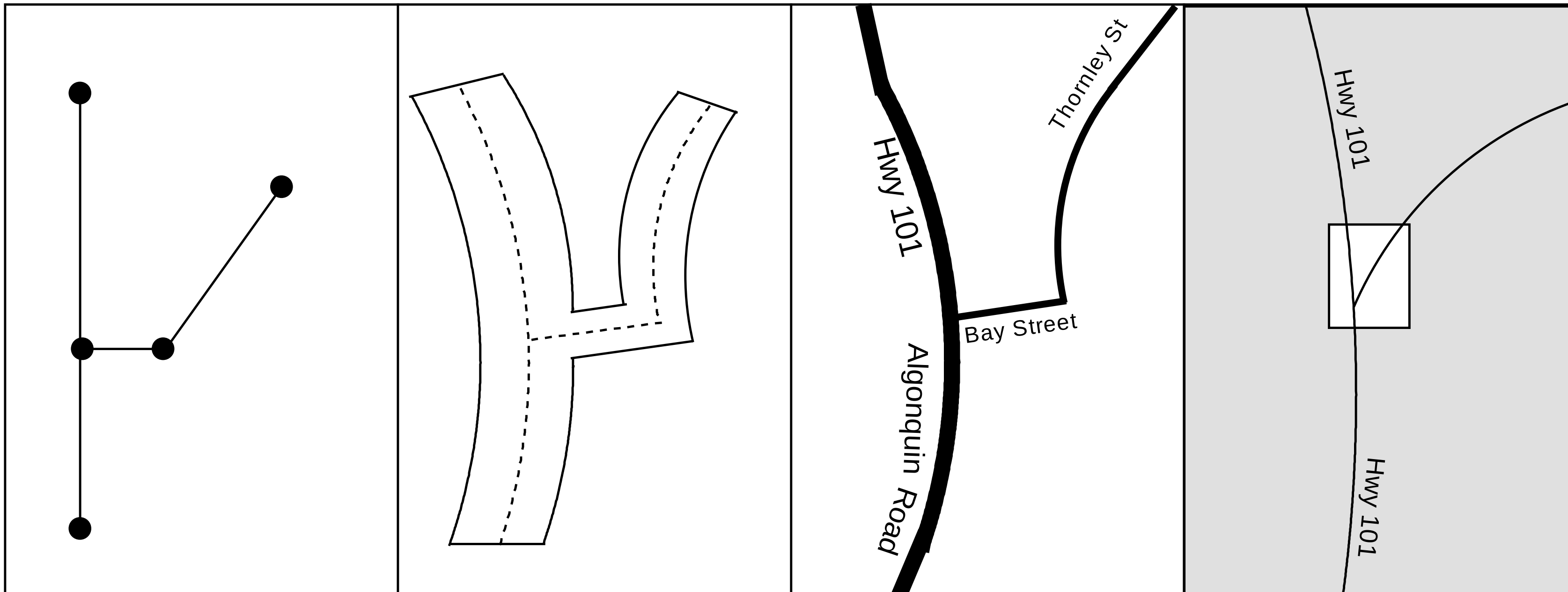


Fig 6-8 A typology of DB heterogeneity



Stored attributes:
Node number [node_num]
Segment number [seg_num]
Flow direction [flow_dir]
Speed limit [spd-lmt]

Stored attributes:
Road name [road_name]
Segment number [segt_no]
Segment length [segt_lngth]
Segment width [segt_wdth]
Number of lanes [lanes]
Surface material [srf_mtl]
Maintenance dates [m_date]

Stored attributes:
Highway number [hwy]_no
Highway name [hwy_name]
Local road name [road]
Hwy/Road classification [class]
Number of lanes [num_lane]
Intersection identification (i_id)

Stored attributes:
Highway number [hwy_num]
Hwy/road classification [clss]

(a) Road Network Database (b) Pavement Management (c) Medium-scale map (d) Small-scale map

Fig 6-9 Conceptualization, formalization & context heterogeneity

3.2 The concept & method of ontology

term is used in the context of spatial DB interoperability as follow :

- a **concept** of using formally defined terminology & vocabulary of real world features
- a **systematic collection & specification** of spatial entities, properties, their relations
- an **emerging approach** to designing spatial DB sys

ontologies are created by consensus among users of data of particular domain, then used

ontology building → semantic mapping → ontology applications

ontology can be documented by various markup language ex. HTML, XML, OIL(ontology interchange language)

can be graphically recorded by UML (E-R diagram)

three approaches to applying ontologies

- global - all data sources are associated w/ one single common domain ontology
- multiple - each data source is related to its own ontology that are logically connected
- hybrid - global + multiple approach

two main role of ontology in supporting DB interoperation

- query translation - mainly the process of translating /mapping heterogeneous field names
- schema integration - mainly combine the schemas of data sources into one global schema

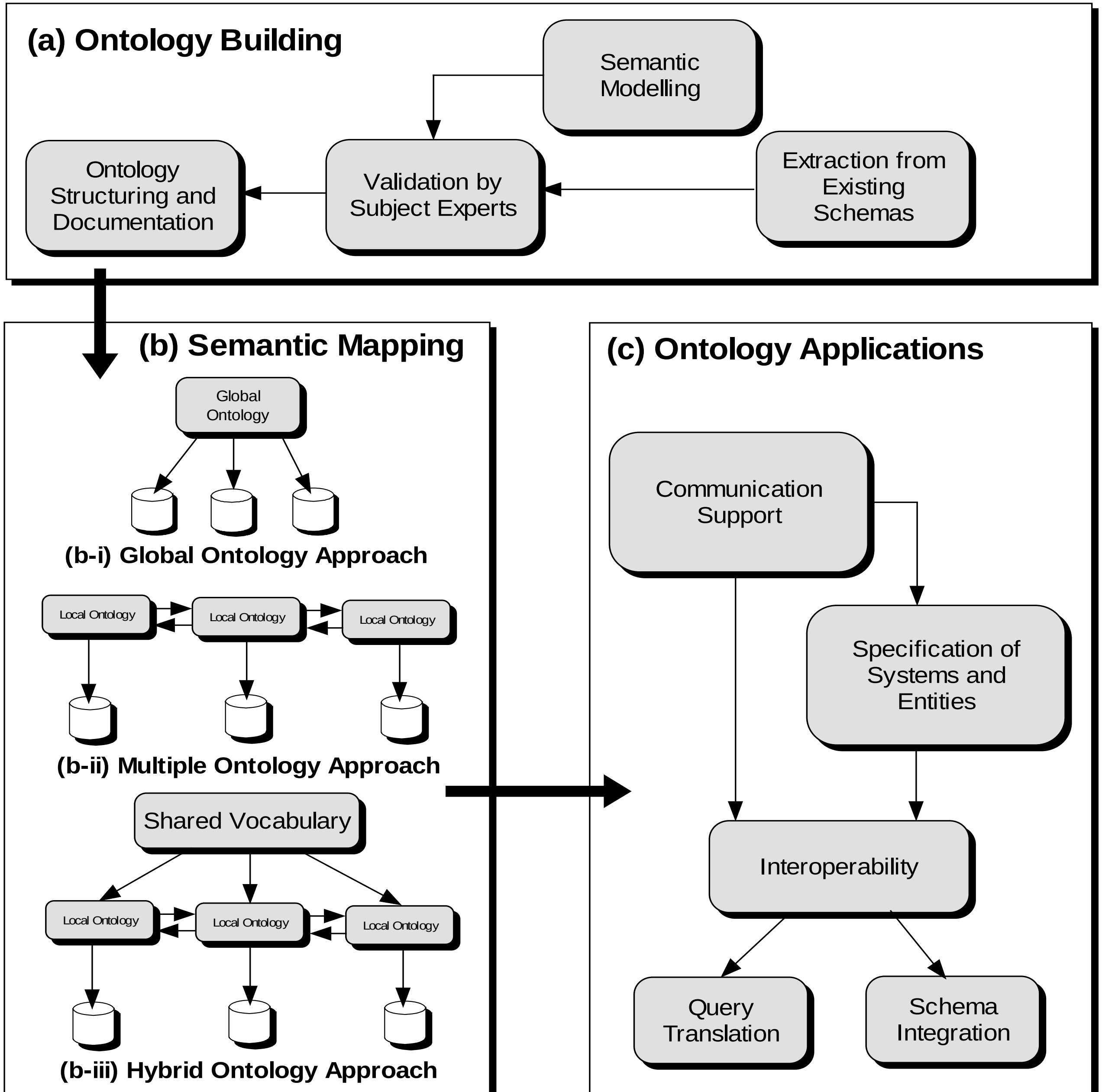


Fig 6-10 Creation & use of ontology in DB sys

3.3 Information mediation

- info mediation is a sort of DB interoperability strategy using **mediator** (middleware medium)
  - mediator : a collection of SW components + DB access optimization rules + catalogue of info
- process used
  - re-writing queries - by optimizing the queries using an execution plan by the DB access rules
  - fragmenting queries to individual data sources + mapping the heterogeneous data field names to their corresponding ontologies using catalogue info
  - dispatching query fragments to target data sources
  - assembling results into a composite response + return to the user interface
- wrapper is also used to manage data source heterogeneity
  - wrapper plays a user interface role - all data become homogeneous to the mediator
- mediator-wrapper architecture represents a higher level of DB interoperation

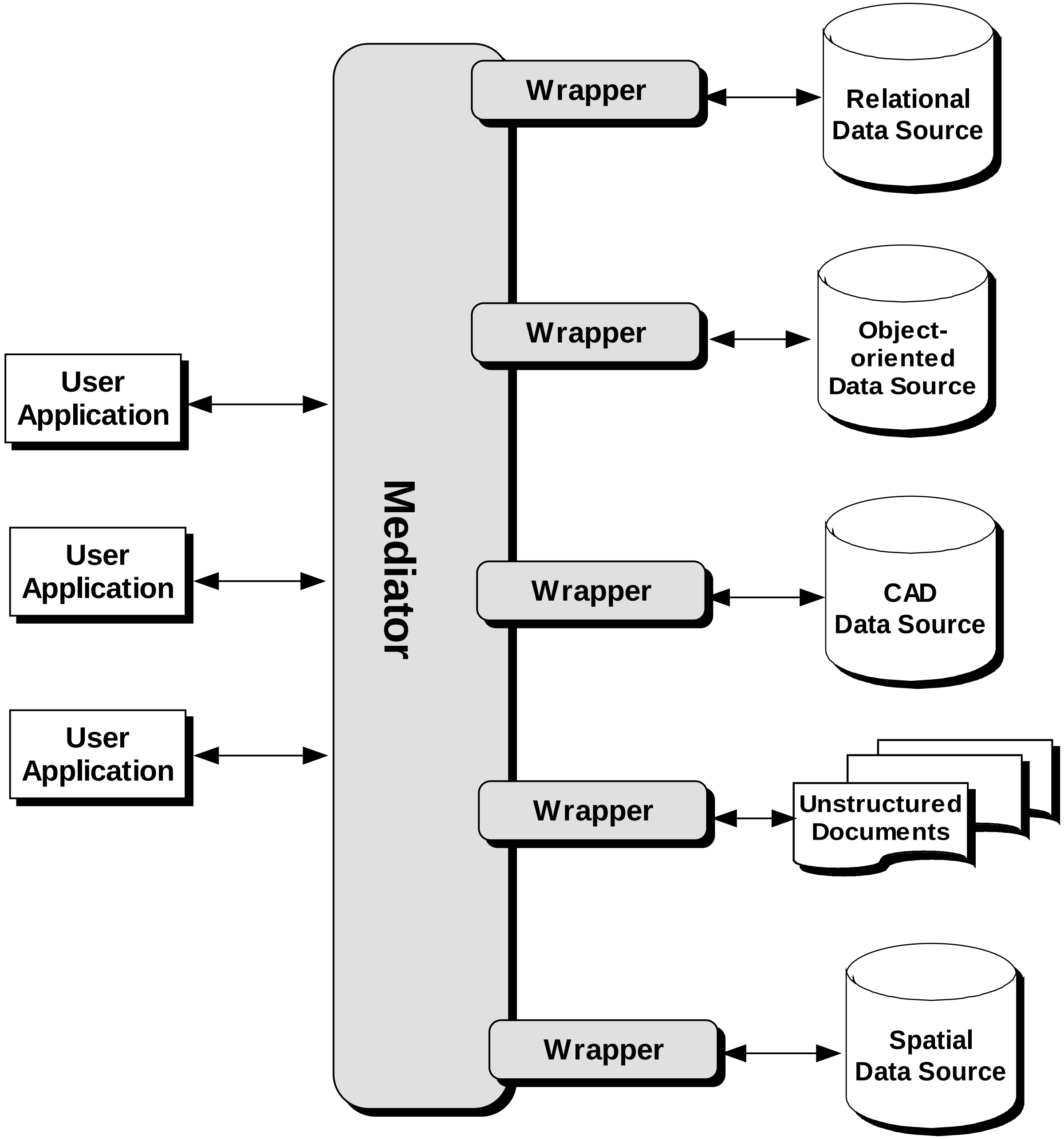


Fig 6-11 The concept of information mediation

4. Data warehousing

two approaches for data sharing - warehousing, federation

warehousing - to merge data physically from several sources

federation - to simultaneously on-line access to multiple data sources

4.1 The definition & characteristics of data warehouse

data warehouse - a special type of DB sys w/ 3 different implementations

: central data repository, enterprise data warehouse, data mart

4.1.1 Central data repository

stores common reference data sets required by different applications

ex. topographic DB maintained by gov't mapping agency

DB is typically stable over time, voluminous, contains common data elements

4.1.2 An enterprise data warehouse

a repository of data derived from operational data sources within an organization

clean data w/o syntactic, structural, semantic heterogeneity → very complex task

4.1.3 A data mart

a subject-specific data warehouse for a particular department / functional unit

smaller size, shorter implementation cycle

can be used as a building blocks for the construction of an enterprise data warehousing

4.1.4 Key features of a data warehouse (especially, of enterprise data warehouse)

four key features of data warehouse(DW) environment

- subject-oriented : DW is organized around major applications of an organization
- integrated : DW is built by integrating data from multiple heterogeneous sources
- time variant : all data sets pertain to a time element
- non-volatile : DW is a separate physical construct from the data sources

4.1.5 Differences between data warehouses & operational DBs

conflicting processing objectives

- DB - for well defined tasks, high availability using indexing & pre-established OLTP
- DW - to perform complex analysis of large legacy data using OLAP

conflicting processing requirements

- DB - for read & write processing, needs concurrent control & recovery mechanisms
- DW - mainly for long read-only processing

conflicting data requirements

- DB - raw business data that may contain errors, incomplete, represent a snapshot of current state
- DW - designed for decision making that data should be clean & pertain to different points in time

4.2 Architecture of a data warehouse

three tier architecture components :

- data warehouse server
  - stores cleaned, transformed data using a multi-dimensional data model
  - communication to the data sources uses DB connectivity standards
  - contains metadata, administration protocols, procedures for management of the DW
- OLAP server
  - maps user queries to the stored data in the DW server
  - two methods of implementing an OLAP server : relational OLAP (ROLAP)
    multi-dimensional OLAP (MOLAP)
- client applications
  - query, reporting tools, data analysis tools, data mining tools

key to DW architecture - multi dimensional data model

- views data in the form of a data cube - represent features from several perspectives / dimensions

commercial vendors - Oracle SDO(Spatial Data Option)+ArcSDE(Spatial Database Engine)
IBM DB2 OLAP Server / DB2 OLAP Miner / DB2 Spatial Extender + ArcSDE

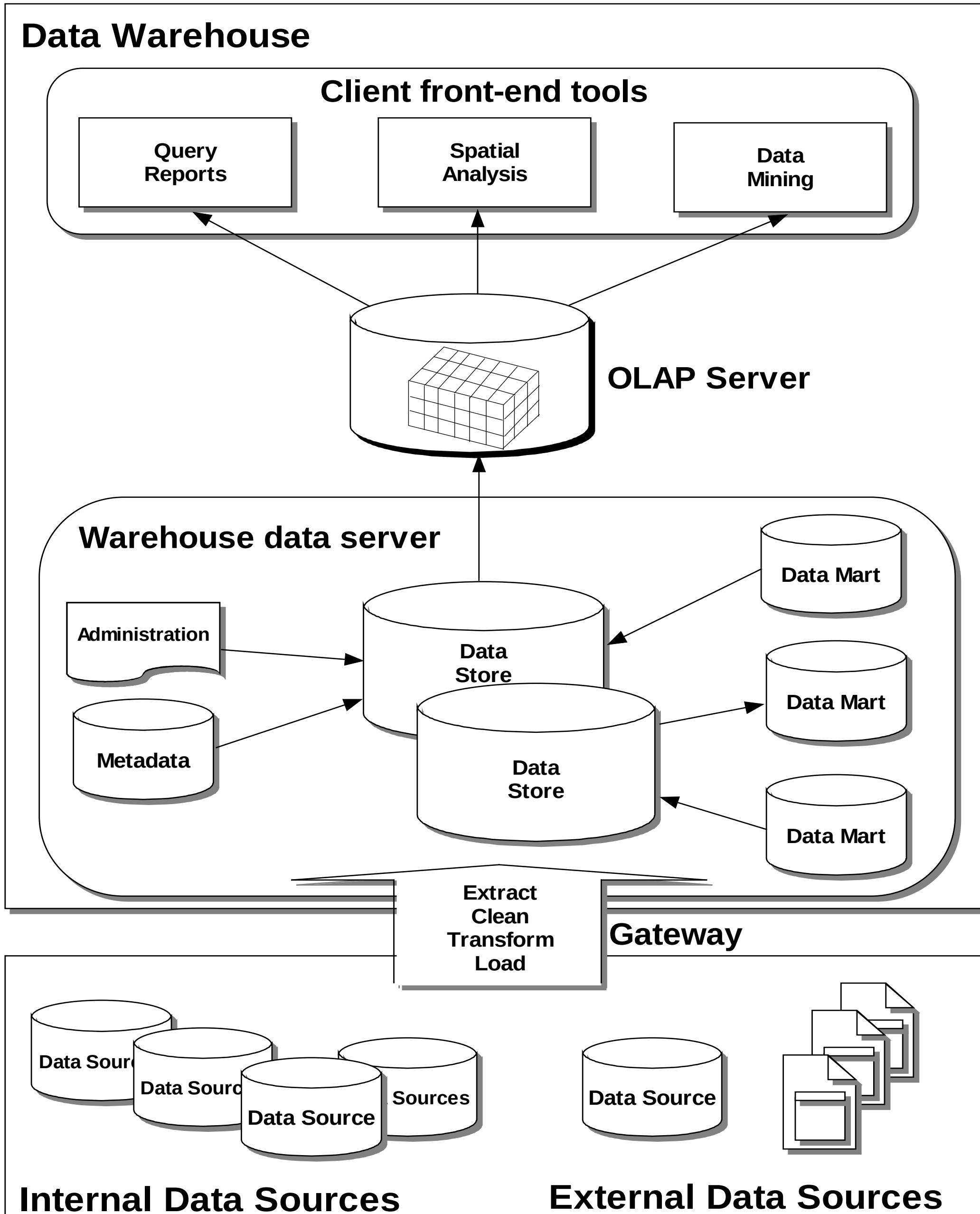


Fig 6-12 The architecture of a data warehouse

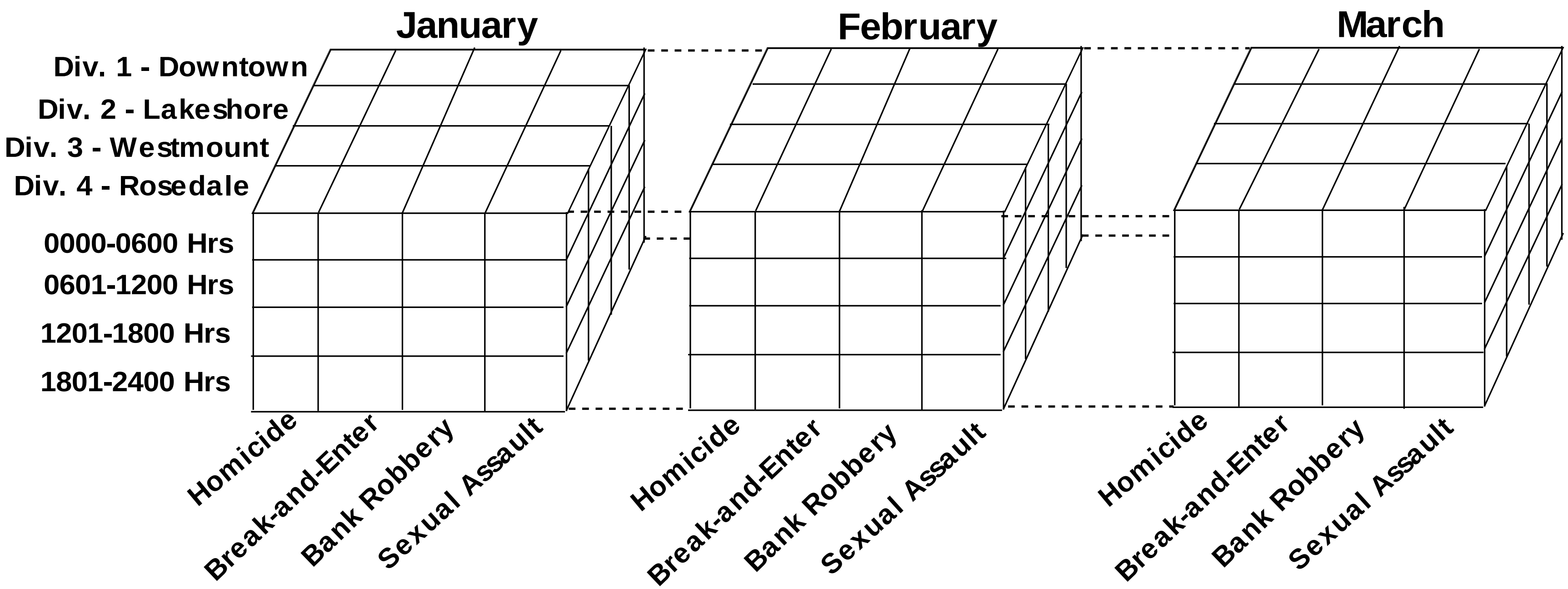


Fig 6-13 A basic multi-dimensional data warehouse model

5 Federated DB systems

5.1 Approaches to DB federation

a. tight federation

makes use of a unified schema as the access interface to member data sources of the federation

schemas of individual DBs are progressively filtered & integrated semantically

b. loose federation

offer a uniform query language (MDBQL : multi-DB query language) which abstracts from the query language of the components

c. mediated federation

used by IBM as a standard solution to data heterogeneity

based on the principles & techniques of info mediation

data sources are federated into a unified system (FDB : federated DB system)

5.2 The architecture of a federated DB system

a. FDB (federated DB system)

control center of the DB federation

communicate w/ client applications using DB connectivity interfaces - ODBC, JDBC, CORBA, DCOM

federated DB server - mediator

b. data sources

structured data in relational / O-relational DBs, spatial data, non/semi-structured data, DW

c. client applications

queries

request for services on remote computers - read & write processes are possible

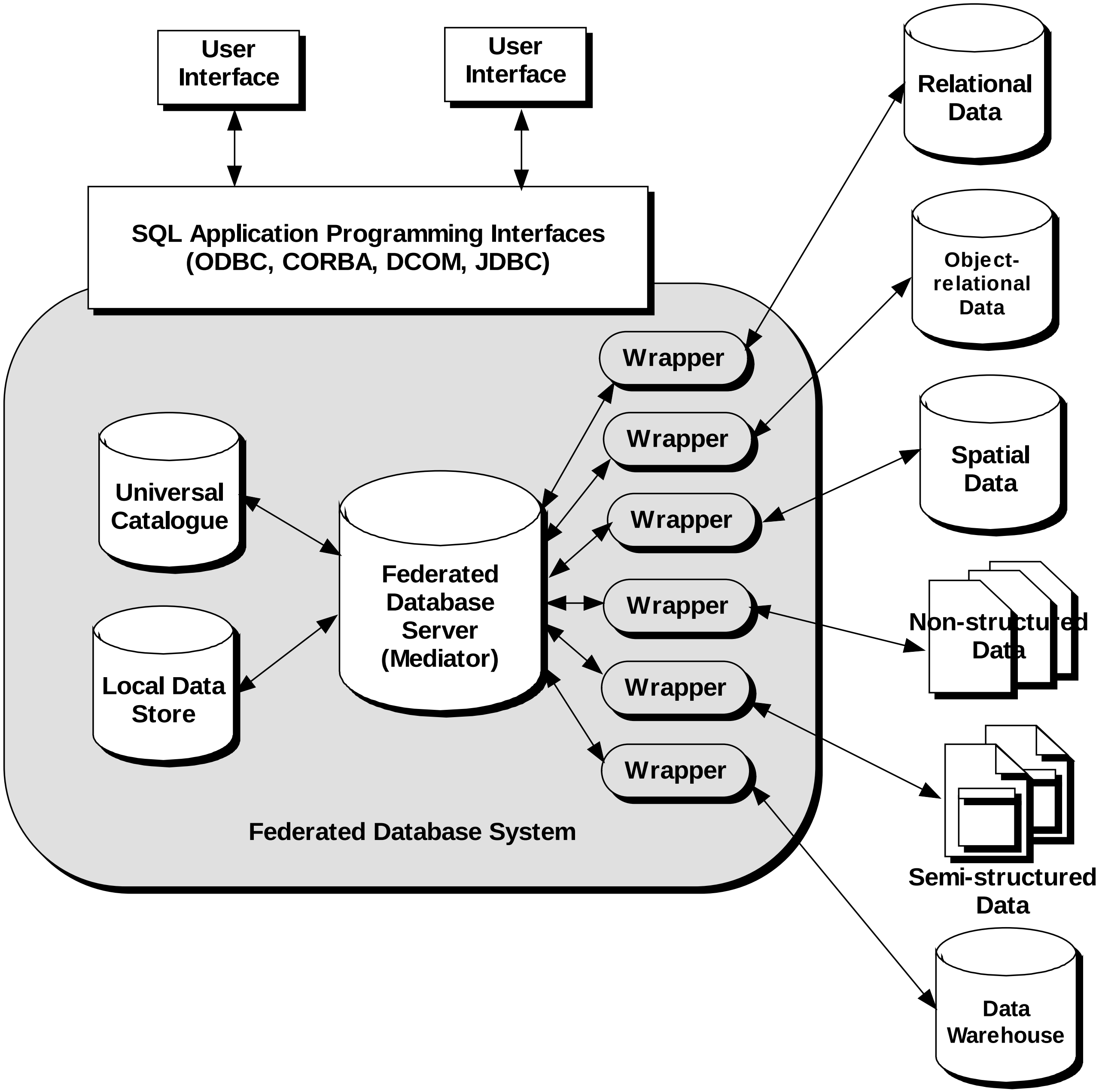


Fig 6-14 Typical DB federation architecture based on the IBM model

5.3 A comparison between data warehouse & DB federation

Table 6-2

Table 6-2. Characteristics of spatial data warehouses and a federated database system

Characteristics	***Data Warehouses***	***Federated Database Systems***
General Description	A collection of subject-oriented data in a well-defined and tightly structured repository	A configuration of geographically distributed, autonomous and heterogeneous data sources and services, communicating using a standard protocol over a network either through schema integration or information mediation
Typical Systems Architecture	Central data server with distributed clients	Distributed data servers with distributed clients
Data Processing Characteristics		
- Local Autonomy	- High	- High
- Concurrent Access	- Easy to control	- Complex to control
- DB Size	- 100 Gigabytes to terabytes	- 100 Megabytes to Gigabytes
- Scalability	- Low	- High
- Modularity	- Low	- High
- Security	- Easy to enforce	- Difficult to enforce
- DP Overhead	- High	- Low
- DP Function	- Subject-oriented	- Application-oriented
- Access/Query	- Mostly read	- Read/write
- View	- Summarised	- Detailed
- Priority	- High performance	- High availability
Network Requirements	Generally high	Generally low
Interoperability Strategy	Pre-computed, data-oriented to merge data physically from several data sources	On-demand, application-oriented strategy to enable simultaneous queries on several data sources on-line
Applicability Scenarios	Relatively small number of structured core data sets, e.g. framework spatial data sets	Large number of distributed and heterogeneous data sources with structured, semi-structured and unstructured data
Application Focus	Subject-oriented data dissemination and OLAP in support of data mining and decision making	Application-oriented distributed services and OLTP in support of business operations
Spatial Database Application	Global, national, state and local reference data, spatial data mining and decision support, multi-dimensional (space, time and attributes) analysis and modelling	Collaborative spatial data analysis and modelling using multi-format and multi-media data, spatial and non-spatial data integration